

APPENDIX A

Notification Reference

The following functions and structures are used with user and application notifications. For further information see the *Windows CE Programmer's Reference*.

Notification Functions

- **PegClearUserNotification**
- **PegGetUserNotificationPreferences**
- **PegHandleAppNotifications**
- **PegRunAppAtEvent**
- **PegRunAppAtTime**
- **PegSetUserNotification**

Notification Structures

- **PEG_USER_NOTIFICATION**

API REFERENCE

The **PegClearUserNotification** function deletes a user notification that was created by a previous call to the function **PegSetUserNotification**.

```
BOOL PegClearUserNotification(    // notify.h
    HANDLE hNotification // handle of notification to delete
);
```

Parameters

hNotification

Identifies the user notification to delete.

Return Values

If the function succeeds, the return value is TRUE. If the function fails, the return value is FALSE.

See also

Windows CE Notifications, **PegSetUserNotification**

The **PegGetUserNotificationPreferences** function queries the user for notification settings by displaying a dialog box showing options that are valid for the current hardware platform.

```
BOOL PegGetUserNotificationPreferences( // notify.h
    HWND hWndParent, // handle of parent window
    PPEG_USER_NOTIFICATION lpNotification // structure with notification settings
);
```

Parameters

hWndParent

Identifies the parent window for the notification settings dialog box.

lpNotification

Points to a **PEG_USER_NOTIFICATION** structure. When calling the function, this structure contains data used to initialize the notification settings dialog box. When the function returns, this structure contains the user's notification settings.

Return Values

If the function succeeds, the return value is TRUE. If the function returns TRUE, the returned settings should be saved, and considered when calling **PegSetUserNotification**.

If the function fails, the return value is FALSE.

See Also

Windows CE Notifications, **PegSetUserNotification**, **PEG_USER_NOTIFICATION**

The **PegHandleAppNotifications** function marks as "handled" all notifications previously registered by the given application. The function turns off the LED and stops vibration (if they were on) only for the given application's events, and removes the taskbar annunciator.

```
BOOL PegHandleAppNotifications( // notify.h
    TCHAR *pwszAppName // name of application whose events are handled
);
```

Parameters

pwszAppName

Points to a null-terminated string that specifies the name of the application whose events are to be marked as "handled".

Return Values

If the function succeeds, the return value is TRUE. If the function fails, the return value is FALSE.

See Also

Windows CE Notifications, **PegGetUserNotification**, **PegSetUserNotification**

The **PegRunAppAtEvent** function starts running an application when the given event occurs.

Windows CE Notes:

Note NOTIFICATION_EVENT_SYSTEM_BOOT is not supported

```
BOOL PegRunAppAtEvent( // notify.h
    TCHAR *pwszAppName, // name of application to run
    LONG IWhichEvent     // event at which the application is to run
);
```

Parameters

pwszAppName

Points to a null-terminated string that specifies the name of the application to be started.

IWhichEvent

Specifies the event at which the application is to be started. This parameter can be one of the following values.

Value Meaning

| | |
|-----------------------------------|--|
| NOTIFICATION_EVENT_NONE | No events—remove all event registrations for this application. |
| NOTIFICATION_EVENT_SYNC_END | When data synchronization finishes. |
| NOTIFICATION_EVENT_ON_AC_POWER | When AC power is connected. |
| NOTIFICATION_EVENT_OFF_AC_POWER | When AC power is disconnected. |
| NOTIFICATION_EVENT_NET_CONNECT | When a network connection is made. |
| NOTIFICATION_EVENT_NET_DISCONNECT | When the network is disconnected. |
| NOTIFICATION_EVENT_DEVICE_CHANGE | When a PCMCIA device is changed. |
| NOTIFICATION_EVENT_IR_DISCOVERED | When an infrared partner is found. |
| NOTIFICATION_EVENT_RS232_DETECTED | When an RS232 connection is made. |
| NOTIFICATION_EVENT_RESTORE_END | When a full device data restore completes. |

Return Values

If the function succeeds, the return value is TRUE. If the function fails, the return value is FALSE.

Remarks

The application is started with a system-defined command line. If there was already an instance of the application running, the new instance must send a private message to the existing instance and then shut down. The command line, which corresponds to the registered event, can be one of the following string constants.

| Constant | Value |
|---------------------------|-------------------------|
| APP_RUN_AT_BOOT | "AppRunAtBoot" |
| APP_RUN_AFTER_SYNC | "AppRunAfterSync" |
| APP_RUN_AT_AC_POWER_ON | "AppRunAtAcPowerOn" |
| APP_RUN_AT_AC_POWER_OFF | "AppRunAtAcPowerOff" |
| APP_RUN_AT_NET_CONNECT | "AppRunAtNetConnect" |
| APP_RUN_AT_NET_DISCONNECT | "AppRunAtNetDisconnect" |
| APP_RUN_AT_DEVICE_CHANGE | "AppRunDeviceChange" |
| APP_RUN_AT_IR_DISCOVERY | "AppRunAtIrDiscovery" |
| APP_RUN_AT_RS232_DETECT | "AppRunAtRs232Detect" |

APP_RUN_AFTER_RESTORE "AppRunAfterRestore"

Remarks

In some cases, the preceding strings are merely the prefix of the command line, and the rest of the command line is used as a parameter.

You should use this function sparingly, because automatically starting an application can confuse the user and cause low-memory conditions on a machine with restricted memory. Ideally, the application should be small and non-intrusive.

See Also

Windows CE Notifications, [PegRunAppAtTime](#), [PegEventHasOccurred](#)

The [PegRunAppAtTime](#) function requests the system to start running the given application at the given time.

```
BOOL PegRunAppAtTime(    // notify.h
    TCHAR *pwszAppName,    // name of application to run
    SYSTEMTIME *lpTime // time when to run the application
);
```

Parameters

pwszAppName

Points to a null-terminated string that specifies the name of the application to be run.

lpTime

Points to a **SYSTEMTIME** structure that specifies the time when the given application is to be run. If this parameter is NULL, the existing run request is deleted and no new request is entered.

Return Values

If the function succeeds, the return value is TRUE. If the function fails, the return value is FALSE.

Remarks

Calling this function replaces any previous run request.

The system passes the APP_RUN_AT_TIME string to the application as the command line. If an instance of the application is already running, the new instance must send a private message to the existing instance and then shut down.

You should use this function sparingly, because automatically starting an application can confuse the user and cause low-memory conditions on a machine with restricted memory. Ideally, the application should be small and non-intrusive.

See Also

Windows CE Notifications, [PegRunAppAtEvent](#)

The **PegSetUserNotification** function creates a new user notification or modifies an existing one.

```
HANDLE PegSetUserNotification(    // notify.h
    HANDLE hNotification, // handle of the notification to overwrite, or zero
    TCHAR *pwszAppName,    // name of application that owns this notification
    SYSTEMTIME *lpTime, // time when the notification is to occur
    PPEG_USER_NOTIFICATION lpUserNotification // contains notification
parameters
);
```

Parameters

hNotification

Identifies the notification to overwrite, or zero to add a new notification.

pwszAppName

Points to a null-terminated string that specifies the name of the application that owns this notification. The system uses the application's primary icon as the taskbar annunciator for the notification. The user can start the application by selecting the annunciator.

lpTime

Points to the **SYSTEMTIME** structure that specifies the time when the notification should occur.

lpUserNotification

Points to the **PEG_USER_NOTIFICATION** structure that describes the events that are to occur when the notification time is reached.

Return Values

If the function succeeds, the return value is the handle of the notification. An application can use the handle to overwrite or delete the notification. The return value is zero if the notification could not be set.

Remarks

The notification occurs at the specified time, without starting the application. The application can specify the notification options, including whether to light the LED, generate a sound, or display a dialog box. However, an application typically uses the **PegGetUserNotificationPreferences** function to allow the user to set the notification options.

The user can start the owning application when the notification occurs. In this case, the system starts a new instance of the application using the **APP_RUN_TO_HANDLE_NOTIFICATION** string as the prefix of the command line, and the notification handle (converted to a string) as the postfix. If another instance of the application is already running, the new instance must pass a private message to the old instance and then shut down.

See Also

Windows CE Notifications, **PegHandleAppNotifications**

The **PEG_USER_NOTIFICATION** structure contains information used to initialize the user notifications settings dialog box, and receives the user's notification preferences entered by way of the dialog box. Also used when setting a user notification.

```
typedef struct UserNotificationType {
    DWORD ActionFlags;
    TCHAR *pwszDialogTitle;
    TCHAR *pwszDialogText;
    TCHAR *pwszSound;
    DWORD nMaxSound;
    DWORD dwReserved;
} PEG_USER_NOTIFICATION, *PPEG_USER_NOTIFICATION;
```

Members

ActionFlags

Specifies the action to take when a notification event occurs. This parameter can be a combination of the following flags.

Value Meaning

PUN_LED Flash the LED.
PUN_VIBRATE Vibrate the device.
PUN_DIALOG Display the user notification dialog box. When this structure is passed to the **PegSetUserNotification** function, the **pwszDialogTitle** and **pwszDialogText** members must provide the title and text of the dialog box.
PUN_SOUND Play the sound specified by the **pwszSound** member. When passed to PSVN, the **pwszSound** member must provide the name of the sound file.
PUN_REPEAT Repeat the **pwszSound** for 10–15 seconds. Only valid if **PUN_SOUND** is set.

Any flag that is not valid on the current hardware platform is ignored.

pwszDialogTitle

Specifies the title of the user notification dialog box. If this parameter is NULL, no dialog is displayed. The **PegGetUserNotificationPreferences** function ignores this member.

pwszDialogText

Specifies the text of the user notification dialog box. If this parameter is NULL, no dialog is displayed. The **PegGetUserNotificationPreferences** function ignores this member.

pwszSound

Points to a buffer that contains the unqualified name of a sound file to play. (The file is assumed to reside in the system media directory.) This parameter is ignored if the **ActionFlags** member does not include the **PUN_SOUND** flag.

nMaxSound

Specifies the maximum length of the string that the **PegGetUserNotificationPreferences** function can copy into the **pwszSound** buffer. Because the string may be a path name in a future release, the buffer must be at least the length derived by the following expression: **PATH_MAX * sizeof(TCHAR)**. This member is ignored by the **PegSetUserNotification** function.

dwReserved

Reserved; must be zero.

Remarks

This structure is passed in the **PegGetUserNotificationPreferences** function. Initial settings are used to populate the dialog. If the function returns TRUE, the returned settings should be saved, and considered when calling **PegSetUserNotification**. Settings for hardware not on the current device will be ignored.

It is also used when calling **PegSetUserNotification**, to describe what should happen when the notification time is reached.

See Also

Windows CE Notifications, PegGetUserNotificationPreferences, PegSetUserNotification

1 **IN THE UNITED STATES PATENT AND TRADEMARK OFFICE**

2 Inventor.....William Vong et al.
3 Applicant Microsoft Corporation
4 Attorney's Docket No. MS1-155US
5 Title: Handheld Computing Device With External Notification System

6 **PATENT ASSIGNMENT**

7 **PARTIES TO THE ASSIGNMENT**

8 Assignor(s):

9 William Vong
10 6511 21st Ave. N.E., Apt. B
11 Seattle, WA 98115

12 Chad Schwitters
13 17615 N.E. 34th Court
14 Redmond, WA 98052-5700

15 Assignee:

16 Microsoft Corporation
17 Corporation of the State of Washington
18 One Microsoft Way
19 Redmond, WA 98052-6399

20 **AGREEMENT**

21 WHEREAS, Assignor(s) are inventor(s) of an invention entitled "Handheld
22 Computing Device With External Notification System," as described and claimed
23 in the specification forming part of an application for United States letters patent
24 executed herewith;

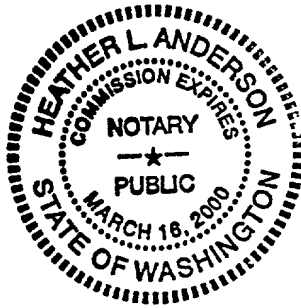
25 WHEREAS, Microsoft, a corporation of the State of Washington having a
 place of business at One Microsoft Way, Redmond, WA 98052, is desirous of
 acquiring the entire right, title and interest in and to the invention and in and to
 any letters patent that may be granted therefor in the United States and in any and
 all foreign countries;

COPY

1 NOW, THEREFORE, in exchange for good and valuable consideration, the
2 receipt of which is hereby acknowledged, Assignor(s) hereby sell, assign and
3 transfer unto Microsoft, the entire right, title and interest in and to said invention,
4 said application and any and all letters patent which may be granted for said
5 invention in the United States of America and its territorial possessions and in any
6 and all foreign countries, and in any and all divisions, reissues and continuations
7 thereof, including the right to file foreign applications directly in the name of
8 Microsoft and to claim priority rights deriving from said United States application
9 to which said foreign applications are entitled by virtue of international
10 convention, treaty or otherwise, said invention, application and all letters patent on
11 said invention to be held and enjoyed by Microsoft and its successors and assigns
12 for their use and benefit and of their successors and assigns as fully and entirely as
13 the same would have been held and enjoyed by Assignor(s) had this assignment,
14 transfer and sale not been made. Assignor(s) hereby authorize and request the
15 Commissioner of Patents and Trademarks to issue all letters patent on said
16 invention to Microsoft. Assignor(s) agree to execute all instruments and
17 documents required for the making and prosecution of applications for United
18 States and foreign letters patent on said invention, for litigation regarding said
19 letters patent, or for the purpose of protecting title to said invention or letters
20 patent therefor.

1
2
3 Date 5/6/97 William Vong
4 William Vong
5 State of Washington)
6 County of King) ss.

7 I certify that I know or have satisfactory evidence William Vong is the person
8 who appeared before me, and said person acknowledged that he signed this
instrument and acknowledged it to be his free and voluntary act for the uses and
purposes mentioned in the instrument.



9 Dated 5/6/97

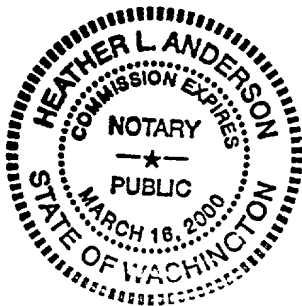
10 Signature of
11 Notary Public Heather L. Anderson

12 My appointment expires March 16, 2000

13 *****

14
15 Date 5/6/97 Chad Schwitters
16 Chad Schwitters
17 State of Washington)
18 County of King) ss.

19 I certify that I know or have satisfactory evidence [inventor name] is the
20 person who appeared before me, and said person acknowledged that he signed this
instrument and acknowledged it to be his free and voluntary act for the uses and
purposes mentioned in the instrument.



21 Dated 5/6/97

22 Signature of
23 Notary Public Heather L. Anderson

24 My appointment expires March 16, 2000
25